Versionierung und Tagging von Images

VERSION:

1.3

STATUS:

in Bearbeitung

AUTOR*IN:

AG „Versionierung und Tagging“ der IG BvC

STAND:

Oktober 2022

IG BvC

INHALT

1 ZIEL DES DOKUMENTES 3
2 GRUNDSÄTZE FÜR DIE ENTWICKLUNG VON CONTAINERLÖSUNGEN 4
3 NOMENKLATUR 5
4 BEGRIFSDEFINITION 6
5 ANFORDERUNG AN DIE VERSIONIERUNG 7
6 DEFINITION TAGGING DER IMAGES 8
6.1 Semantische Versionierung 8
6.2 Erweiterung der semantischen Versionierung 8
6.3 Beispiele für semantische Versionierung 9
6.3.1 Semantische Versionierung bei Standard-Image: 9
6.3.1.1 Grund-Image Tag: *9*
6.3.1.2 Runtime-Image Tag: *9*
6.3.2 Semantische Versionierung bei Applikation-Image (Empfehlung): 9
6.4 Erhöhung des Image-Tags 10
6.5 Image-Tagging für Entwickler 13
6.5.1 Rolling Tag bei Standard-Images 13
6.5.1.1 Rolling Tag Grund-Images *13*
6.5.1.2 Rolling Tag Runtime-Images *15*
6.5.2 Rolling Tag bei Applikation Image (Empfehlung) 17
6.6 Digest im Image-Tag 18
6.7 Zusatz zum Image-Tag 18
6.8 Label von Images 19
7 ANWENDUNGSBEREICH DER IMAGE-TAGS 20

1 ZIEL DES DOKUMENTES

Mit diesem Dokument beschreibt die IG Betrieb von Container eine Empfehlung zur Standardisierung der Versionierung und Tagging von Container Images. Dieses Dokument richtet sich an Softwarelieferant und Softwarebetreiber.

2 GRUNDSÄTZE FÜR DIE ENTWICKLUNG VON CONTAINERLÖSUNGEN

Die IG Betrieb von Containern (IG BvC) ist ein Zusammenschluss von Datenzentralen, Softwarelieferanten und Organisationen der Öffentlichen Verwaltung.

Ziel der IG BvC ist die Schaffung von Standards für das Deployment von Fachverfahren in Container-Umgebungen sowie für den Betrieb von Container-Plattformen zur Herstellung einer Kompatibilität zwischen den Rechenzentren. Die Schaffung einer herstellerunabhängigen Systematik ist das Grundverständnis der IG BvC.

Die Schaffung und Umsetzung von konkreten Möglichkeiten der Koppelung und gemeinsamen Nutzung von Container-Plattformen zwischen den Rechenzentren von Bund, Ländern und Kommunen ist ein wesentlicher Grundsatz für die Zusammenarbeit.

Die Ausarbeitungen der AG Cloud und Digitale Souveränität werden bei der Ausarbeitung der Standards genutzt. Im Gegenzug werden die Ausarbeitungen der IG BvC für die Nutzung durch die AG bereitgestellt.

Die Ausarbeitungen des BSI-Bausteins SYS.1.6 Container (Community Draft) werden als Grundlage genutzt. Als grundlegender Standard für die Betrachtungen wird Kubernetes gesetzt. Sämtliche Betrachtungen werden plattformunabhängig ausgeführt um eine Abhängigkeit von Produkten und Herstellern weitestgehend auszuschließen.

3 NOMENKLATUR

Die Verwendung und Bedeutung der modalen Hilfsverben (alternativ Modalverben) ist in der DIN-Norm 820-2 oder RFC2119 geregelt. Ergänzend zu diesen Regeln werden die Modalverben bzw. Schlüsselwörter „MUSS“, „SOLL“, „IST ZU“, „DARF NICHT“, „SOLLTE“, „SOLLTE NICHT“, „KANN“ und „DARF“ wie folgt verwendet:

- „**MUSS**“, „**IST ZU**“, „**DARF NUR**“ weisen auf eine absolut zu erfüllende Anforderung hin (uneingeschränkte Anforderung).
- „**DARF NICHT**“, „**DARF KEIN**“ ist die Negierung einer „MUSS“-Anforderung und stellt ein Verbot dar (uneingeschränktes Verbot).
- „**SOLLTE**“, „**SOLL**“ oder das entsprechende Adjektiv „EMPFOHLEN“ bedeuten, dass eine Anforderung normalerweise erfüllt werden muss, es aber Gründe geben kann, dies doch nicht zu tun. Dies muss aber sorgfältig abgewogen und stichhaltig begründet werden.
- „**SOLLTE NICHT**“ oder das korrespondierende Adjektiv „NICHT EMPFOHLEN“ sind die entsprechenden Negierungen und bedeuten, dass etwas normalerweise nicht getan werden sollte, es aber Gründe gibt, dies doch zu tun. Dies muss aber sorgfältig abgewogen und stichhaltig begründet werden.
- „**KANN**“, „**DARF**“ zeigt eine Option an.

Folgende Definitionen für Rollen werden in der IG Betrieb von Containern verwendet:

- Der **Softwarelieferant** ist eine Organisation (juristische Person, Community), welche Softwarereleases bereitstellt.
  Wenn möglich setzt sie die Anforderungen des Software- und Plattformbetreibers um.
- Der **Plattformbetreiber** betreibt die IT-Infrastruktur Kontext IaaS und PaaS für die Containerumgebung im Rechenzentrum und stellt Mittel zur manuellen und/oder automatischen Orchestrierung bereit.
- Der **Softwarebetreiber** verantwortet den Betrieb einer Anwendung / eines Service entsprechend vertraglicher Verpflichtungen gegenüber Kunde/Auftraggeber und managed die Orchestrierung von Containern.
  Wenn möglich, stimmt er die Anforderungen an den Betrieb der Software mit dem Softwarelieferanten ab.
  Es ist das Bindeglied zwischen Plattformbetreiber und Softwarelieferant.

4 BEGRIFSDEFINITION

Image-Name: Ist eine Eindeutige Bezeichnung eines Images, welche die Funktion und Inhalt Beschreibt.

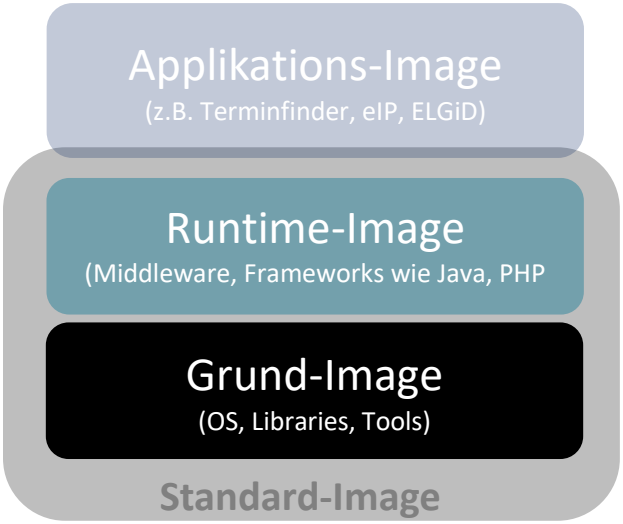
Grund-Image: Das Grund-Image wird von scratch erstellt oder vom Images Hersteller bezogen und enthält die notwendigen Komponenten auf Betriebssystemebene. Es ist eine Minimaldistribution eines Anbieters, z. B. Linux-Distribution.

Runtime-Image: Das Runtime-Image erweitert ein Grund-Images um Komponenten der Middleware und Frameworks, damit Anwendungen lauffähig werden.

Applikation-Image: Das Applikation-Image erweitert ein Grund-Image oder Runtime-Image um Komponenten zum Betrieb einer Anwendung.

Entwickler-Version: Entwickler-Version eines Images sind Dev-Images, die für die Herstellung einer Anwendung zusätzliche Werkzeuge beinhalten, die im Rahmen der produktiven Bereitstellung nicht mehr benötigt werden und daher für den Produktivbetrieb nicht mehr erlaubt sind.

Folgender Aufbau weiß eine Konsistenz von Applikation-Image einer Fachanwendung auf:



Eine Vollständige Definition eines Standardimages ist in der Ausarbeitung „PoC – Standardisierung der Standard-Images“ zu finden.

5 ANFORDERUNG AN DIE VERSIONIERUNG

- **Softwarelieferant SOLL** für Sicherheitsupdates in den elementaren Bestandteilen im verwendeten Standard-Image (z.B. libc) keine neue Versionsnummer in seinem Anwendungsimage vergeben.
- **Softwarelieferant MUSS** die Granularität der automatischen Updates über die Version des Standard-Images steuern können (Beispiel: php:8 vs. php:8.1 vs. php 8.1.0).
- **Softwarelieferant MUSS** die aktuellste Version eines Images Erkennbar bezeichnen.
- **Softwarebetreiber MUSS** Anwendungsimages anhand einer eindeutigen Versionsbezeichnung unterscheiden können.
- **Softwarebetreiber MUSS** sicherstellen, dass Updates mittels Version und Tags erkannt und permanent, anlassbezogen sowie zeitnah installiert werden.
- **Softwarebetreiber**:
    - SYS.1.6.A6: Es **DARF NUR** eindeutige Versionsnummern angegeben werden.

- Grundimage Ersteller (**Softwarelieferant**):
    - Es müssen gehärtete Images erstellt werden (z.B. ohne Shell und Paketmanager)
        - ⇨ Damit ungehärtete Entwickler-Images von produktiv nutzbaren Images zu unterscheiden, MÜSSEN Entwickler-Images im Image-Name oder Image-Tag kenntlich gemacht werden. Z.B. – unhardened

- Bei freigegebenen Images aus öffentlichen Registries, welche in der OpenCoDE Plattform hochgeladen und bereitgestellt werden, **SOLLTE** das bereits unterliegende Versionierungsschema aus der öffentlichen Registry unverändert übernommen werden.
- Das aktuellste Image **MUSS** erkennbar bezeichnet sein.
- Das stabile/funktionierende Image **MUSS** erkennbar Bezeichnet sein

6 DEFINITION TAGGING DER IMAGES

6.1 Semantische Versionierung

Bei der Versionierung von Container-Images soll die semantische Versionierung Anwendung finden. Die Version ist dreistellig in der Form *<Major>.<Minor>.<Patch>* wobei gilt:

- Die erste Stelle bezeichnet die **Major-Version**. Die Major-Version wird bei inkompatiblen Änderungen angehoben.
- Die zweite Stelle bezeichnet die **Minor-Version**. Die Minor-Version wird bei kompatiblen Änderungen angehoben.
- Die dritte Stelle bezeichnet die **Patch-Version**. Die Patch-Version wird bei Fehlerbehebungen angehoben.

Versionsbezeichner sind dabei fest einem Stand der Quellen für die Erstellung eines Containerimage zugeordnet und sind unveränderbar, werden also einmal gesetzt und nicht wiederverwendet bzw. auf einen anderen Softwarestand verschoben. Dadurch wird gewährleistet, dass Softwarezustände (Versionen) unveränderbar und mit einer eindeutigen Bezeichnung abgelegt werden.

6.2 Erweiterung der semantischen Versionierung

Die Semantische Versionierung mit **<Major>.<Minor>.<Patch>** wird mit einer **Build-Nummer** erweitert. Mit Hilfe der Build-Nummer werden neue Versionen des Images, aufgrund fortlaufender automatischer Erstellprozesse, z.B. eines aktualisierten Grundimages, unterschieden.

Durch das Hochzählen der Build-Nummer symbolisiert der höchste nummerische Wert das neueste Image.

6.3 Beispiele für semantische Versionierung

6.3.1 Semantische Versionierung bei Standard-Image:

6.3.1.1 Grund-Image Tag:

<Grund-Image-Name>:<major>.<minor>.<patch>-<build>

Fallbeispiel für ein Grund-Image:
Grund-Image: Debian11:

Grund-Image Tag:

Debian11:1.0.0-BuildNummer

6.3.1.2 Runtime-Image Tag:

<Runtime-Image-Name><ru_major>-<Grund-Image-Name><gr_major>:<major>.<minor>.<patch>-<build>

Dabei gilt „ru_major“ als Majorversion der Runtime und „gr_major“ als Majorversion des Grund-Image.

Fallbeispiel für ein Standard-Image:
Grund-Image: Debian11:
Runtime: Java11

Standard-Image Tag:

Java11-Debian11:1.0.0-BuildNummer

6.3.2 Semantische Versionierung bei Applikation-Image (Empfehlung):

Es wurde gemeinsam in der Interessengemeinschaft für den Betrieb von Container festgelegt, dass das Tagging der Applikation-Images nur eine Empfehlung an den Softwarelieferanten darstellt soll.

==Empfehlung==

<Applikation-Image-Name>:<major>.<minor>.<patch>-build

Fallbeispiel Semantische Versionierung eines Applikation-Image:

TerminfinderSH:

Terminfindersh:2.0.1-12345

BundesMessenger:

BundesMessenger:1.1.2-12345

6.4 Erhöhung des Image-Tags

Das Erhöhen des Image-Tags folgt einer bestimmen Einordnung der jeweiligen Änderungen des Images. Auf folgender Grundlage werden die Elemente der Versionsnummer von MAJOR.MINOR.PATCH-BUILD erhöht:

1. **MAJOR** wird erhöht, wenn API-inkompatible Änderungen veröffentlicht werden,
2. **MINOR** wird erhöht, wenn neue Funktionalitäten, welche kompatibel zur bisherigen API sind, veröffentlicht werden,
3. **PATCH** wird erhöht, wenn die Änderungen ausschließlich API-kompatible Bugfixes umfassen, und
4. **BUILD** wird erhöht, wenn das Image durch z.B. Major, Minor, Patch oder durch ein anderes Ereignis (Applikation unabhängige Veränderungen am Standard-Image) neu gebaut werden muss.

   ⇨ **Die Erhöhung der Versionsnummer MAJOR.MINOR.PATCH-BUILD im Standard-Image ist abhängig von Änderungen des Runtime- und Grund-Images.**

   ⇨ **Die Erhöhung der Versionsnummer MAJOR.MINOR.PATCH-BUILD im Applikation-Image ist abhängig von Änderungen des Runtime-, Grund-Image und der Applikation.**

Verdeutlichung der Erhöhung des Image-Tags am Beispiel:
Update eines Standard-Images

Standard-Image-Tag	1.2.3-12345
Major-Tag	1
Minor-Tag	1.2
Patch-Tag	1.2.3
Build-Tag	12345
Latest-Tag	latest

Major-Update:

Standard-Image-Tag	1.2.3-12345	2.2.3-12346
Major-Tag		2
Minor-Tag		2.2
Patch-Tag		2.2.3
Build-Tag		12346
Latest-Tag		latest

Minor Update:

Standard-Image-Tag	1.2.3-12345	2.2.3-12345	2.3.3-12346
Major-Tag		2	2
Minor-Tag		2.2	2.3
Patch-Tag		2.2.3	2.3.3
Build-Tag		12345	12346
Latest-Tag			latest

Patch Update:

Standard-Image-Tag	1.2.3-12345	2.2.3-12345	2.3.3-12346	2.3.4-12347
Major-Tag		2	2	2
Minor-Tag		2.2	2.3	2.3
Patch-Tag		2.2.3	2.3.3	2.3.4
Build-Tag		12345	12346	12347
Latest-Tag				latest

Veränderung des Images ohne Major, Minor oder Patch Update:

Standard-Image-Tag	1.2.3-12345	2.2.3-12345	2.3.3-12346	2.3.4-12347	2.3.4-12348
Major-Tag		2	2	2	2
Minor-Tag		2.2	2.3	2.3	2.3
Patch-Tag		2.2.3	2.3.3	2.3.4	2.3.4
Build-Tag		12345	12346	12347	12348
Latest-Tag					latest

6.5 Image-Tagging für Entwickler

Das Versionierungskonzept von Container-Images sieht vor, dass die Image-Tags rollierend auf einen neueren Softwarestand zeigen können. Die Verwendung von Rolling-Image-Tags als Versionsbezeichner können für Entwickler für nichtöffentliche Images eingesetzt werden und wird im Folgenden erklärt.

1. **Rolling Tag latest**
   Der neuste Entwicklungsstand ist mit dem Tag latest markiert. Dieses Tag ist veränderbar und wird bei neu veröffentlichten Container-Images verschoben.

2. **Rolling Tag <Major>**
   Das aktuellste Container-Image eines Major-Versions-Zweiges wird mit einem Tag der Form <*Major*> versehen. Dieses Tag ist veränderbar und wird bei neu veröffentlichten Container-Images dieses Versionszweiges verschoben. Verschiebungen können hier durch neu veröffentlichte kompatible Änderungen oder Fehlerbehebungen ausgelöst werden.

3. **Rolling Tag <Major>.<Minor>**
   Das aktuellste Container-Image eines Minor-Versions-Zweiges wird mit einem Tag der Form <*Major*>.<*Minor*> versehen. Dieses Tag ist veränderbar und wird bei neu veröffentlichten Container-Images dieses Versionszweiges verschoben. Verschiebungen können hier durch neu veröffentlichte Fehlerbehebungen ausgelöst werden.

Der Aspekt des Rolling Tag ist ein Paradigmenwechsel zur herkömmlichen Versionierung von Softwareartefakten. Die Verschiebung der Tags ermöglicht es beim Betrieb von Container ein festes Versionslevel einzubinden und dennoch darunterliegende Änderungen automatisch zu erhalten. Dabei können die Rolling Tags zusätzlich zum normalen Tagging angewendet werden.

Das beschriebene Versionierungskonzept ist im Container-Umfeld etabliert und wird von populären Softwareherstellern praktiziert. Als Beispiele können hier die Dockerfiles und die Versionshistorie von postgres, traefik, nginx, mongodb, python, etc. herangezogen werden.

6.5.1 Rolling Tag bei Standard-Images

6.5.1.1 Rolling Tag Grund-Images

```
<Grund-Image-Name>:latest
<Grund-Image-Name>:<Major>
<Grund-Image-Name>:<Major>.<Minor>
<Grund-Image-Name>:<Major>.<Minor>-Build
```

Beispiel für Rolling Tags am Grund-Image Debian11:

Rolling Tag lastest:	Debian11:latest
Rolling Tag <Major>:	Debian11:1
Rolling tag <Major>.<Minor>:	Debian11:1.0

Beispiel: Update eines Grund-Images

Grund-Image-Tag	1.1.1-12345
Rolling-Major-Tag	1
Rolling-Minor-Tag	1.1
Latest-Tag	latest

Update Patch:

Grund-Image-Tag	1.1.1-12345	1.1.2-12345
Rolling-Major-Tag		1
Rolling-Minor-Tag		1.1
Latest-Tag		latest

Update Minor:

Grund-Image-Tag	1.1.1-12345	1.1.2-12345	1.2.1-12346
Rolling-Major-Tag			1
Rolling-Minor-Tag		1.1	1.2
Latest-Tag			latest

Update Major:

Grund-Image-Tag	1.1.1-12345	1.1.2-12345	1.2.1-12346	2.2.0-12347
Rolling-Major-Tag			1	2
Rolling-Minor-Tag		1.1	1.2	2.2
Latest-Tag				latest

6.5.1.2 Rolling Tag Runtime-Images

<Runtime-Image-Name>.<Grund-Image-Name>:latest
<Runtime-Image-Name>.<Grund-Image-Name>:<Major>
<Runtime-Image-Name>.<Grund-Image-Name>:<Major>.<Minor>
<Runtime-Image-Name>.<Grund-Image-Name>:<Major>.<Minor>-<Build>

Beispiel für Rolling Tags am Runtime-Image Java11 mit Grund Image Debian:

Rolling Tag lastest:	Java11-Debian11:latest
Rolling Tag <Major>:	Java11-Debian11:1
Rolling Tag <Major>.<Minor>:	Java11-Debian11:1.0
Rolling Tag <Major>.<Minor>-<Build>:	Java11-Dabian11:1.0-12345

Es wird auf den Patch Tag verzichtet, da dies bereits in der „normalen“ Versionierung vorhanden ist.

Beispiel: Update eines Runtime-Images

Runtime-Image-Tag	1.1.1-12345
Rolling-Major-Tag	1
Rolling-Minor-Tag	1.1
Patch-Tag	1.1.1
Latest-Tag	latest

Update Patch:

Runtime-Image-Tag	1.1.1-12345	1.1.2-12346
Rolling-Major-Tag		1
Rolling-Minor-Tag		1.1
Patch-Tag	1.1.1	1.1.2
Latest-Tag		latest

Update Minor:

Runtime-Image-Tag	1.1.1-12345	1.1.2-12346	1.2.2-12347
Rolling-Major-Tag			1
Rolling-Minor-Tag		1.1	1.2
Patch-Tag	1.1.1	1.1.2	1.2.2
Latest-Tag			latest

Update Major:

Runtime-Image-Tag	1.1.1-12345	1.1.2-12346	1.2.2-12347	2.2.2-12348
Rolling-Major-Tag			1	2
Rolling-Minor-Tag		1.1	1.2	2.2
Patch-Tag	1.1.1	1.1.2	1.2.2	2.2.2
Latest-Tag				latest

6.5.2 Rolling Tag bei Applikation Image (Empfehlung)

Beispiel: Update eines Applikation-Images

Applikation-Image-Tag	1.1.1-12345
Rolling-Major-Tag	1
Rolling-Minor-Tag	1.1
Patch-Tag	1.1.1
Latest-Tag	latest

Patch Update:

Applikation-Image-Tag	1.1.1-12345	1.1.2-12346
Rolling-Major-Tag		1
Rolling-Minor-Tag		1.1
Patch-Tag	1.1.1	1.1.2
Latest-Tag		latest

Minor Update

Applikation-Image-Tag	1.1.1-12345	1.1.2-12346	1.2.2-12347
Rolling-Major-Tag			1
Rolling-Minor-Tag		1.1	1.2
Patch-Tag	1.1.1	1.1.2	1.2.2
Latest-Tag			latest

Major Update

Applikation-Image-Tag	1.1.1-12345	1.1.2-12346	1.2.2-12347	2.2.2-12348
Rolling-Major-Tag			1	2
Rolling-Minor-Tag		1.1	1.2	2.2
Patch-Tag	1.1.1	1.1.2	1.2.2	2.2.2
Latest-Tag				latest

Beispiel an einer Applikation „Terminfinder S-H“:

- Terminfinder S-H Version 2

Rolling Tag lastest:	TerminfinderSH:latest
Rolling Tag <Major>:	TerminfinderSH:2
Rolling tag <Major>.<Minor>:	TerminfinderSH:2.0

6.6 Digest im Image-Tag

Digest finden im Image-Tagging, aufgrund des geringen Informationsgehaltes, keine Verwendung und werden nicht benötigt.

6.7 Zusatz zum Image-Tag

Der Zusatz zum Image-Tag sind Bezeichnungen um das Image näher zu beschreiben. Folgender Zusatz zum Image-Tag ist definiert:

Unhardened: Werden jene Images getaggt, welche in der openCoDE Plattform bereitgestellt werden und nicht die BSI Vorgaben entsprechen. Das Image kann für das Bauen weiterer Images oder zu Debugging Zwecken eingesetzt werden.
Bezeichner im Image: -unhardened

Prerelease: Prerelease werden jene Images getaggt, welche eine Vorabversion der Anwendung enthält, jedoch noch nicht für den Produktiv Betrieb vorgehsehen ist.
Bezeichner im Image: -prerelease

6.8 Label von Images

Images müssen mit zusätzlichen Metadaten, unabhängig vom Image Tage und Image Version, ergänzt werden. Mit Hilfe von Labels werden die zusätzlichen Metadaten an das Image gesetzt. Folgende Information sind in Labels in Form von „Annotation Keys“ an die Images zu setzten:

- Version der Anwendung im Image (**org.opencontainers.image.version)**
- Ersteller des Images (**org.opencontainers.image.authors)**
- URL auf eigenes Repository (**org.opencontainers.image.url)**
- Welche Lizenzierung/Lizenzmodel wird benötigt/angewendet (**org.opencontainers.image.licenses)**
- Datum und Zeit wann das Image entstanden ist (**org.opencontainers.image.created)**

Bezug zu: https://github.com/opencontainers/image-spec/blob/main/annotations.md#pre-defined-annotation-keys

Es können weitere „Annotation Keys“ zur besseren Beschreibung des Images verwendet werden. Die aufgeführte Liste ist für die Bereitstellung der Images zwingend aufzuführen und zu beschreiben. An einem Beispiel soll verdeutlich werden, welche Informationen als Metadaten an das Image ergänzt werden kann.

```
LABEL summary="My WebSite"
description="This is my Site"
io.k8s.description="This is my Site"
io.k8s.display-name="My WebSite"
io.openshift.expose-services="8080:http”
io.openshift.tags="MyWebSite.org-1.1"
name="siegl/MyWebSite.org”
version="1.1"
maintainer="Andreas Wesemann <andreas.wesemann@lfst.bayern.de>"
```

7 ANWENDUNGSBEREICH DER IMAGE-TAGS

Die Versionierung soll bei folgenden Images angewendet werden:

- Standard Images (Grund + Runtime Images)
- Applikation Images (Empfehlung)

➔ **Die Versionierung soll für alle Images gelten, die öffentlich in der OpenCoDE-Plattform zu Verfügung gestellt werden.**

ÄNDERUNGSKONTROLLE

Version	Datum	Bearbeiter	Status, Änderungsgrund	Seiten
0.1	08.06.2022		Erstversion	Alle
1.0	12.09.2022		Endversion	Alle
1.1	21.09.2022		Kleine Anpassungen	3, 6, 7, 8, 12, 13
1.3	06.10.2022		Hinzufügen von Fallbeispielen, Hinzufügen der Erläuterung des Build Tag als Erweiterung zur semantischen Versionierung, Ergänzung Fallbeispiel Image Label	Alle
1.4	26.10.2022		Endversion	
1.5	01.02.2023		Update/Änderung	9, 10, 11, 12, 13, 14,15, 16, 17, 18, 19